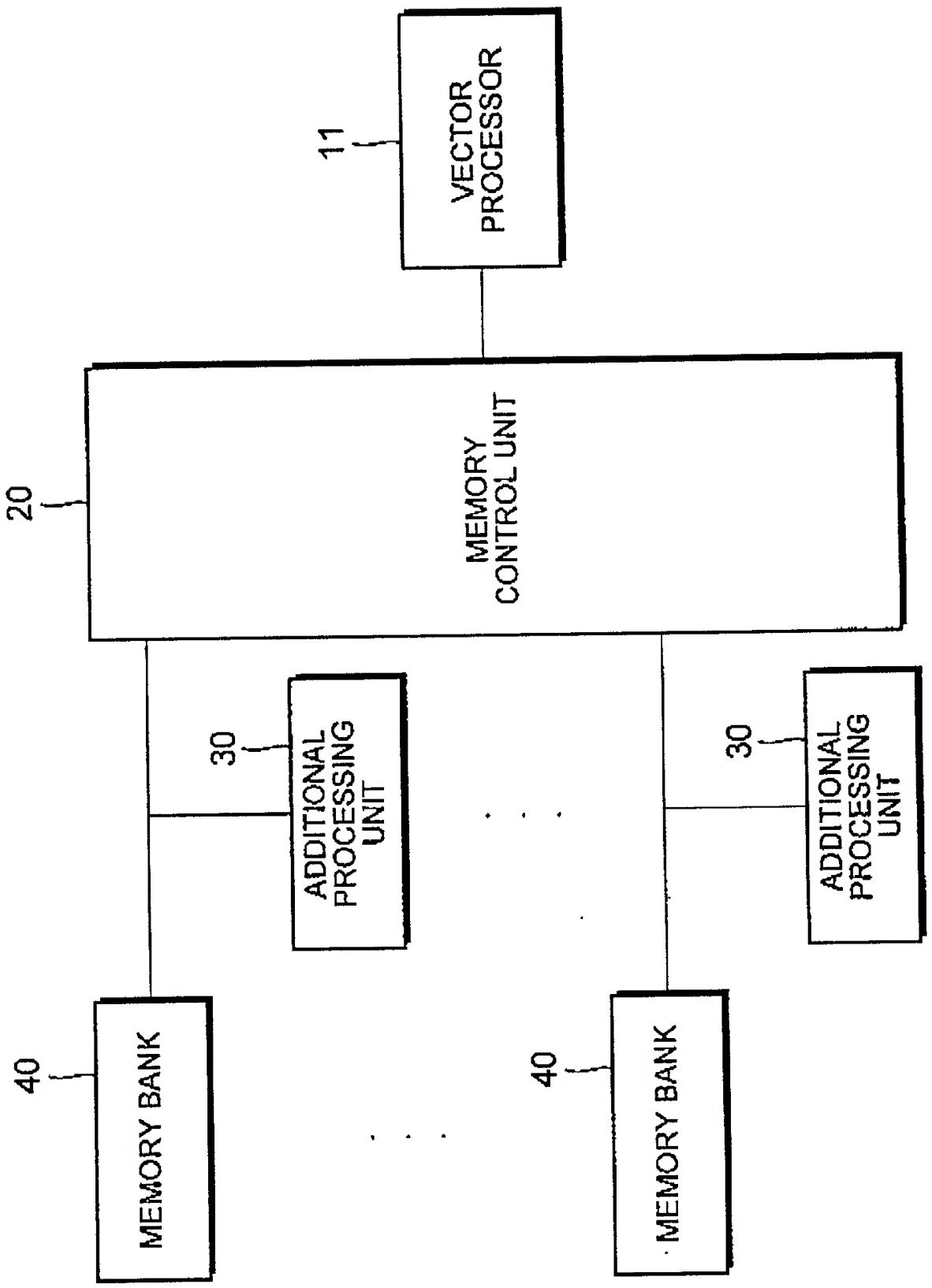
11
VECTOR PROCESSOR
20
MEMORY CONTROL UNIT
30
ADDITIONAL PROCESSING UNIT
40
MEMORY BANK
30
ADDITIONAL PROCESSING UNIT
40
MEMORY BANK

FIG. 1

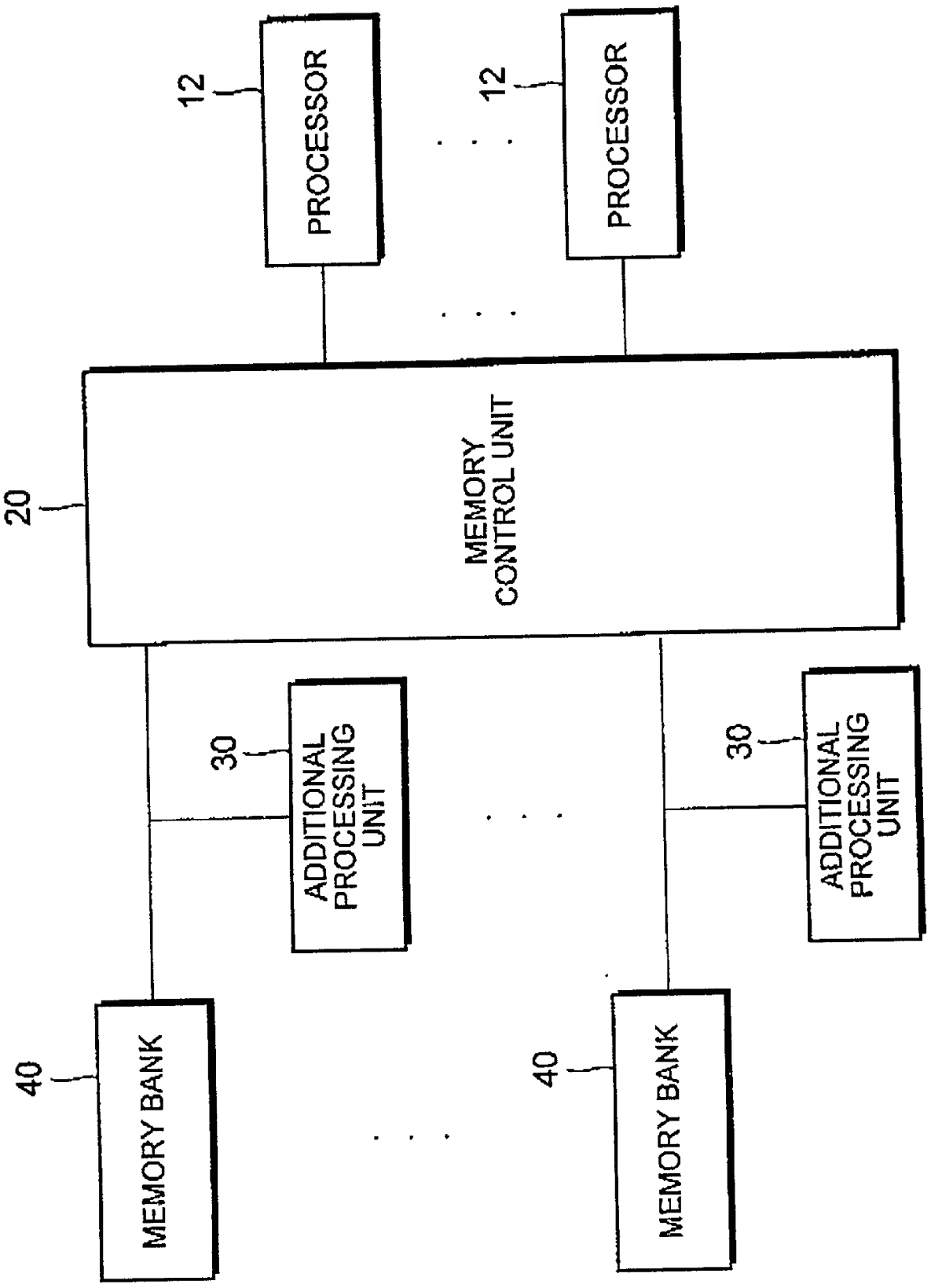
12
PROCESSOR
12
PROCESSOR
20
MEMORY CONTROL UNIT
30
ADDITIONAL PROCESSING UNIT
30
ADDITIONAL PROCESSING UNIT
40
MEMORY BANK
40
MEMORY BANK

FIG. 2

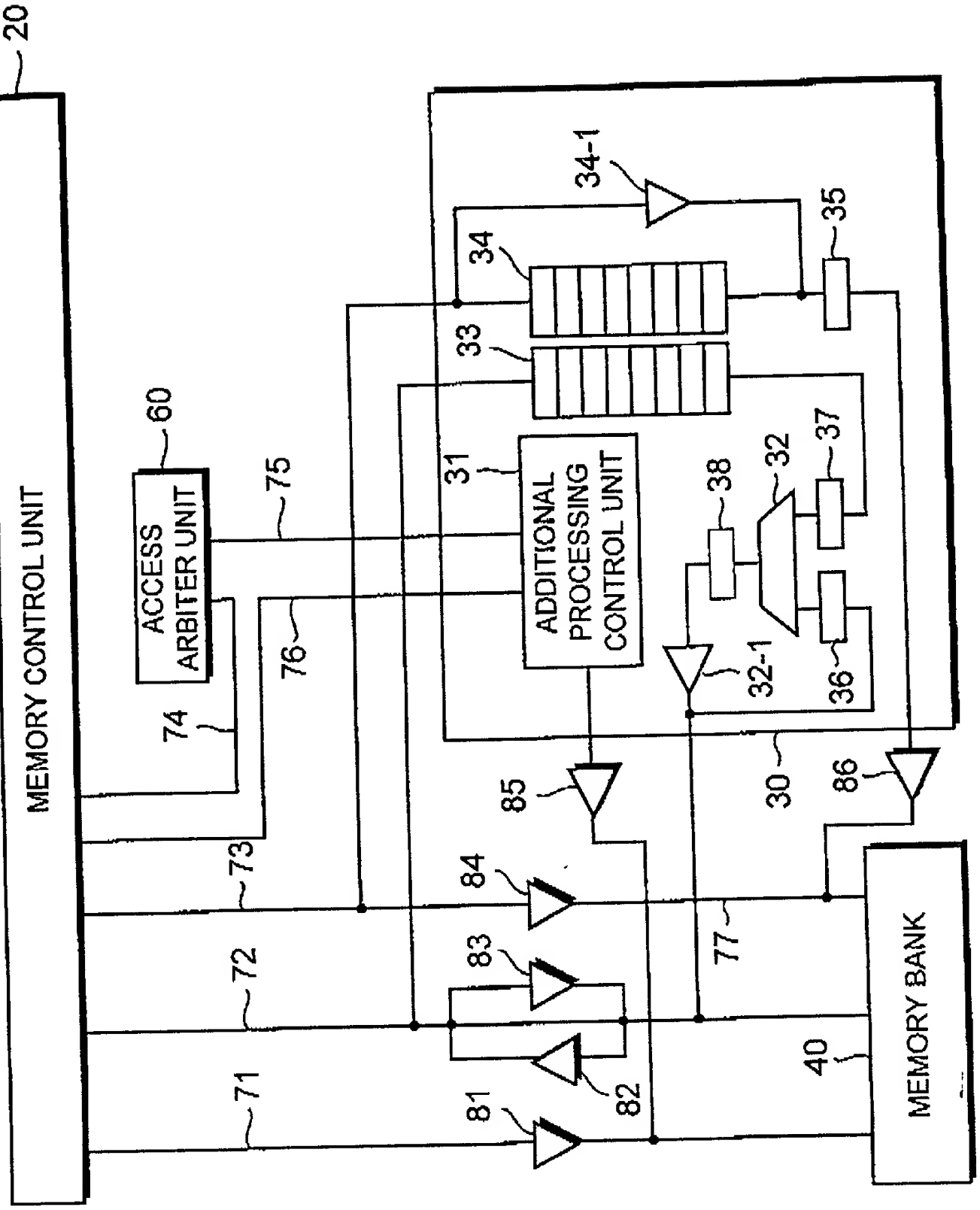
20
MEMORY CONTROL UNIT
60
ACCESS ARBITER UNIT
75
76
74
73
72
71
31
ADDITIONAL PROCESSING CONTROL UNIT
34-1
34
33
35
38
32
37
32-1
36
85
84
83
82
81
30
86
77
40
MEMORY BANK

FIG. 3

$$Y + = f\ (a,b,c, \cdot\ \cdot\ \cdot\ )$$

FIG. 4

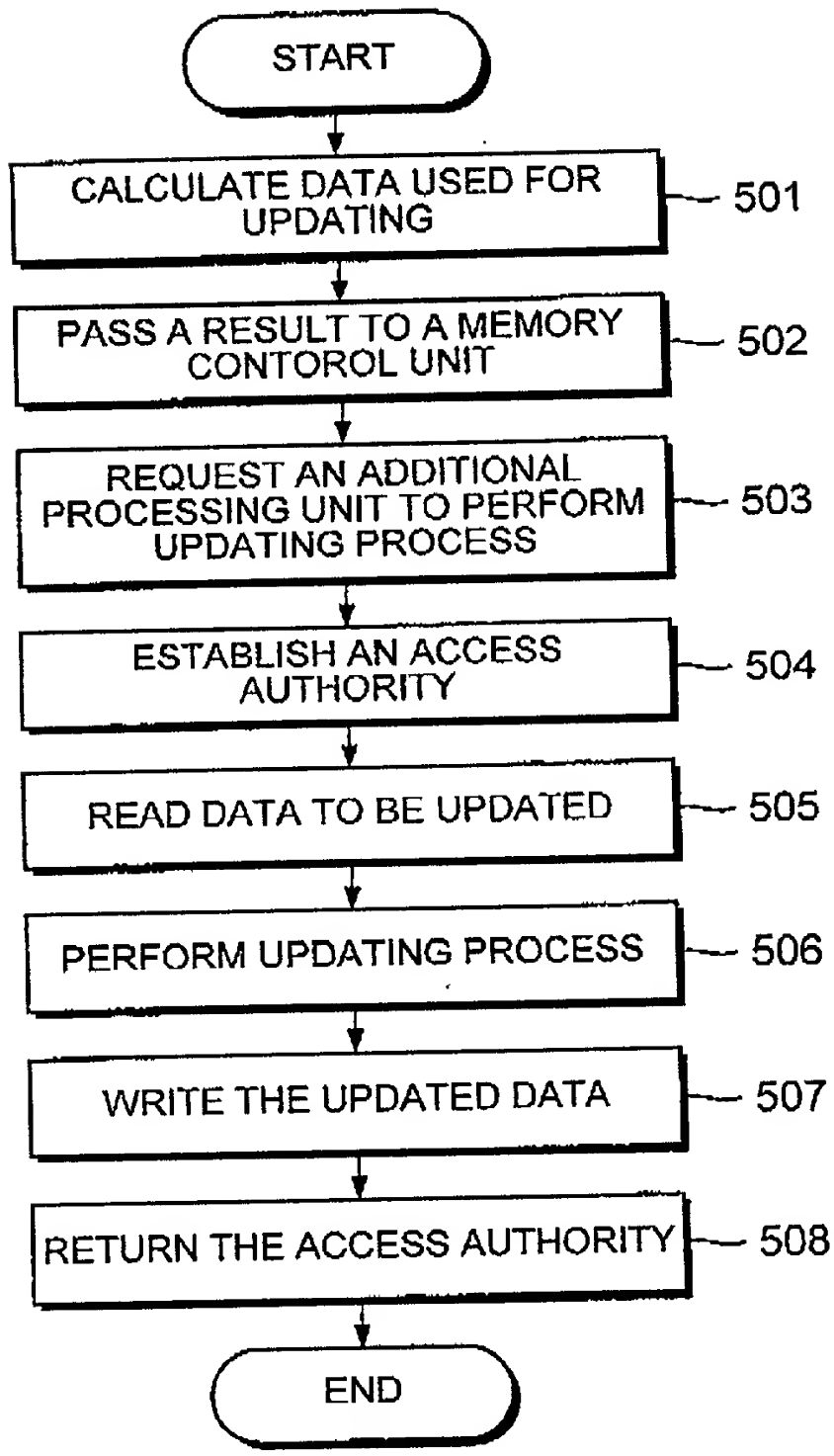

FIG. 5

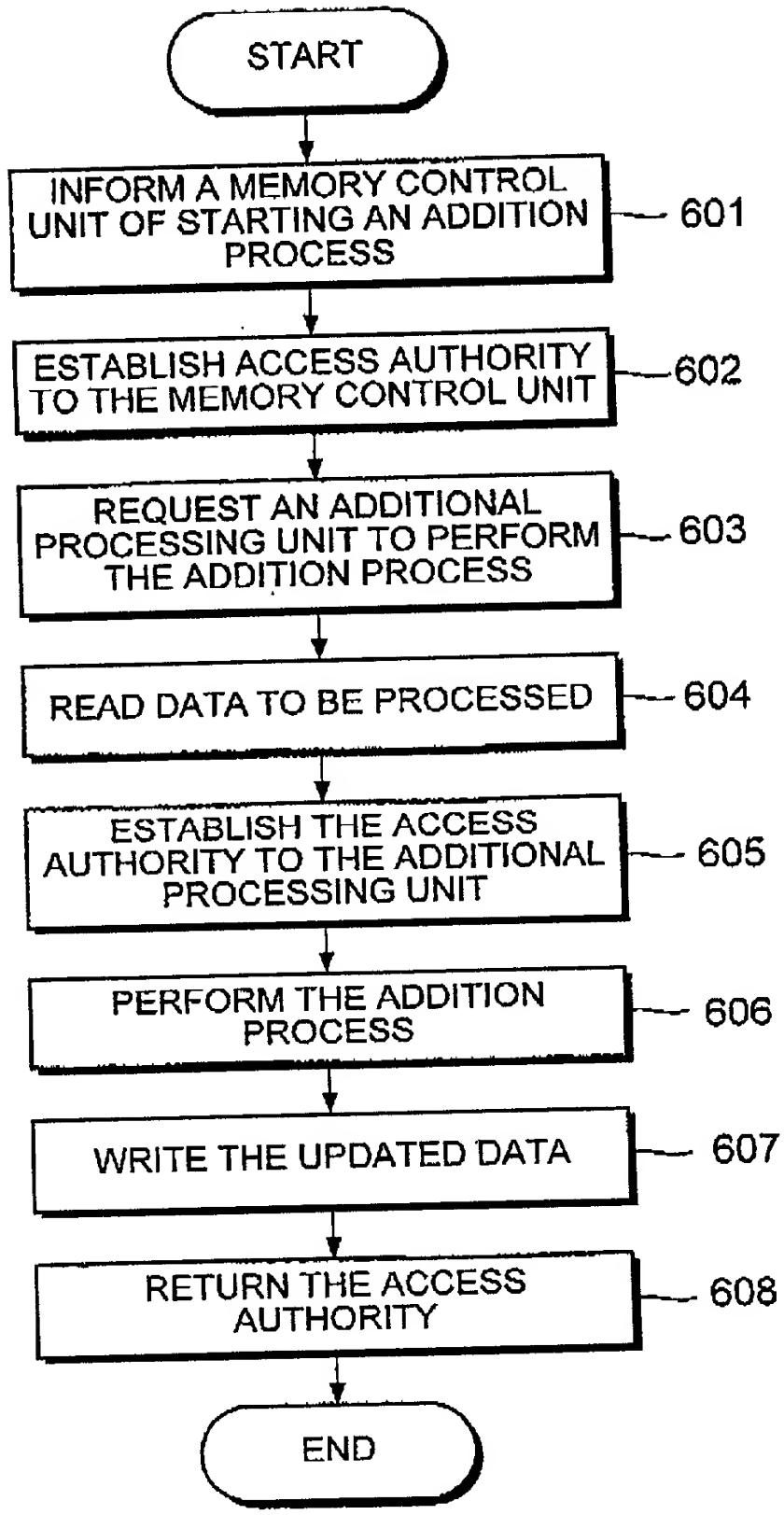
START
INFORM A MEMORY CONTROL UNIT OF STARTING AN ADDITION PROCESS
601
ESTABLISH ACCESS AUTHORITY TO THE MEMORY CONTROL UNIT
602
REQUEST AN ADDITIONAL PROCESSING UNIT TO PERFORM THE ADDITION PROCESS
603
READ DATA TO BE PROCESSED
604
ESTABLISH THE ACCESS AUTHORITY TO THE ADDITIONAL PROCESSING UNIT
605
PERFORM THE ADDITION PROCESS
606
WRITE THE UPDATED DATA
607
RETURN THE ACCESS AUTHORITY
608
END

FIG. 6

$$y = \begin{pmatrix} y \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \end{pmatrix} + \begin{bmatrix} A \\ 6\,5\,0\,0\,0 \\ 0\,0\,0\,0\,0 \\ 4\,0\,0\,0\,0 \\ 0\,0\,0\,0\,0 \\ 0\,0\,0\,0\,3 \end{bmatrix} \begin{pmatrix} x \\ 2 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

FIG. 7

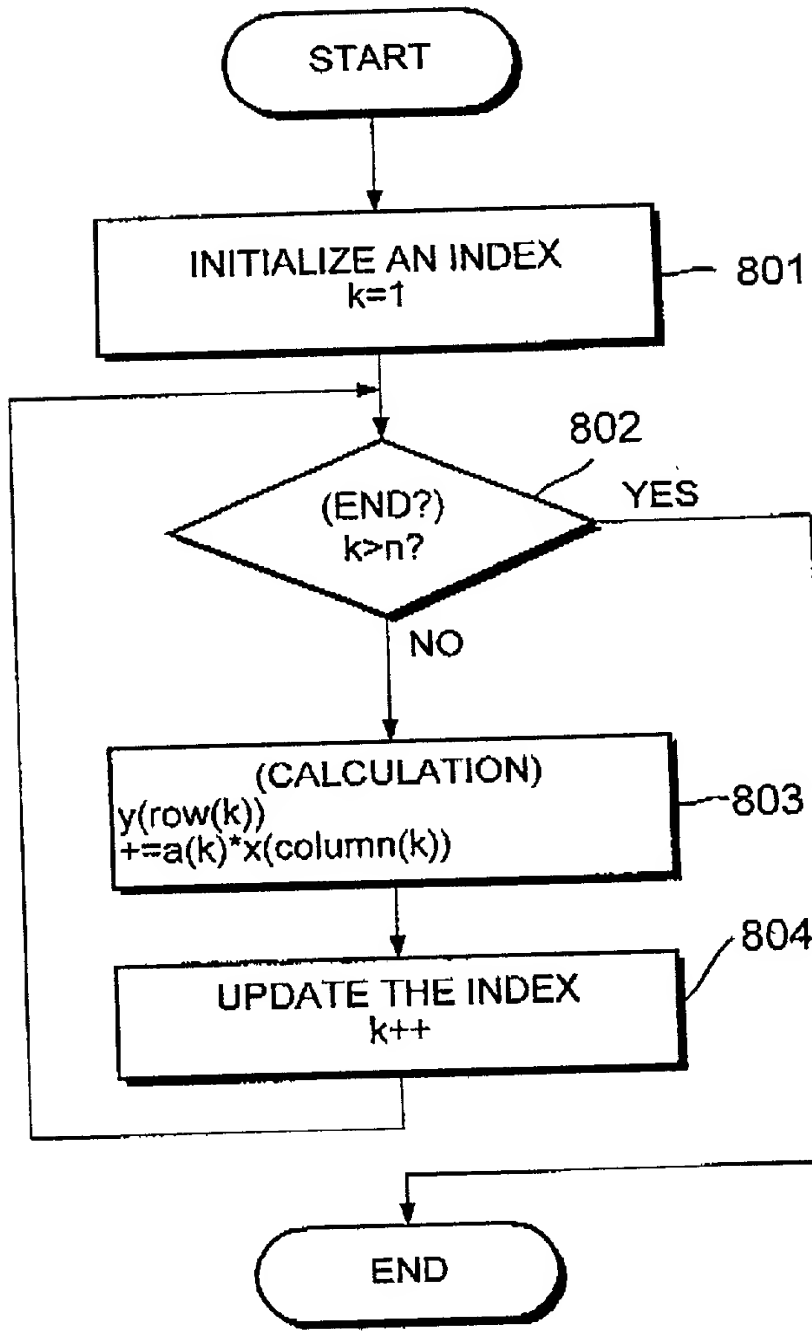

FIG. 8

k	a(k)	column(k)	x(column(k))	a(k)*x(column(k))
1	6	1	2	12
2	5	2	0	0
3	4	1	2	8
4	3	5	1	3

FIG. 9

k	a(k)*x(column(k))	row(k)	y(row(k))	y(row(k))+a(k)*x(column(k))
1	12	1	7	19
2	0	1	19	19
3	8	3	9	17
4	3	5	11	14

FIG. 10

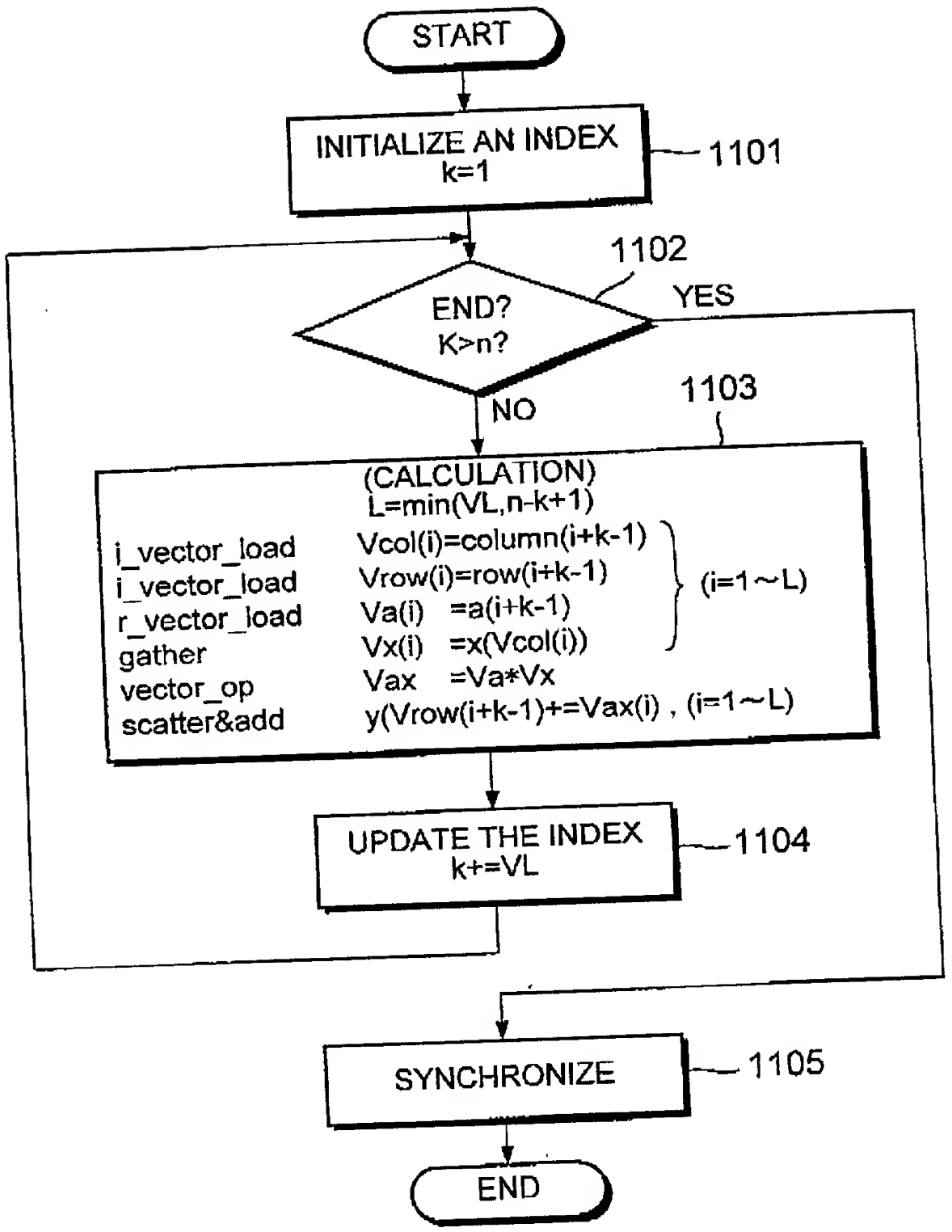
START
INITIALIZE AN INDEX
k=1
1101
1102
END?
K>n?
YES
NO
1103
(CALCULATION)
L=min(VL,n-k+1)
i_vector_load Vcol(i)=column(i+k-1)
i_vector_load Vrow(i)=row(i+k-1)
r_vector_load Va(i) =a(i+k-1)
gather Vx(i) =x(Vcol(i))
vector_op Vax =Va*Vx
scatter&add y(Vrow(i+k-1)+=Vax(i) , (i=1~L)
(i=1~L)
UPDATE THE INDEX
k+=VL
1104
SYNCHRONIZE
1105
END

FIG. 11

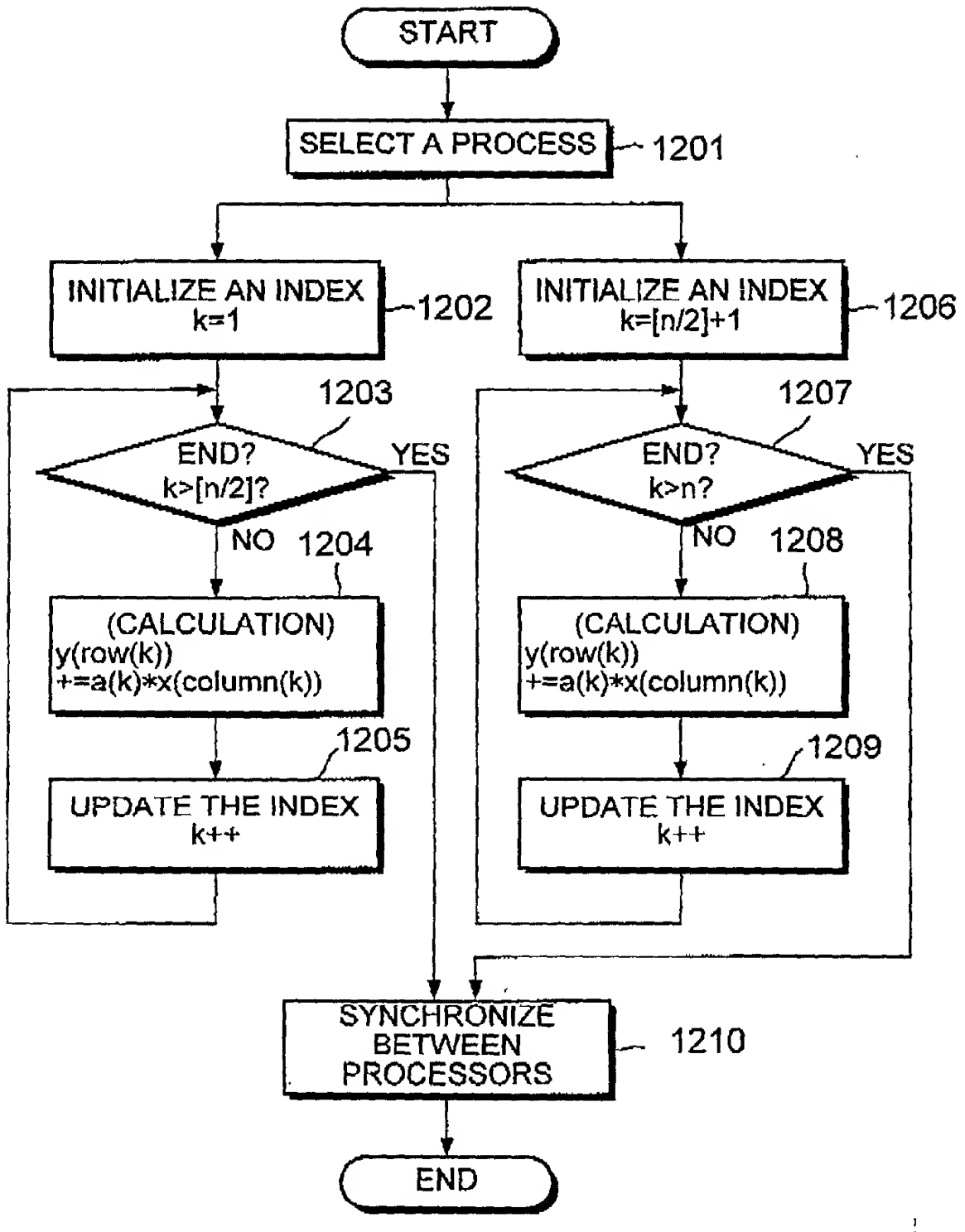
START
SELECT A PROCESS
1201
INITIALIZE AN INDEX
k=1
1202
INITIALIZE AN INDEX
k=[n/2]+1
1206
1203
END?
k>[n/2]?
YES
NO
1204
(CALCULATION)
y(row(k))
+=a(k)*x(column(k))
1205
UPDATE THE INDEX
k++
1207
END?
k>n?
YES
NO
1208
(CALCULATION)
y(row(k))
+=a(k)*x(column(k))
1209
UPDATE THE INDEX
k++
SYNCHRONIZE
BETWEEN
PROCESSORS
1210
END

FIG. 12

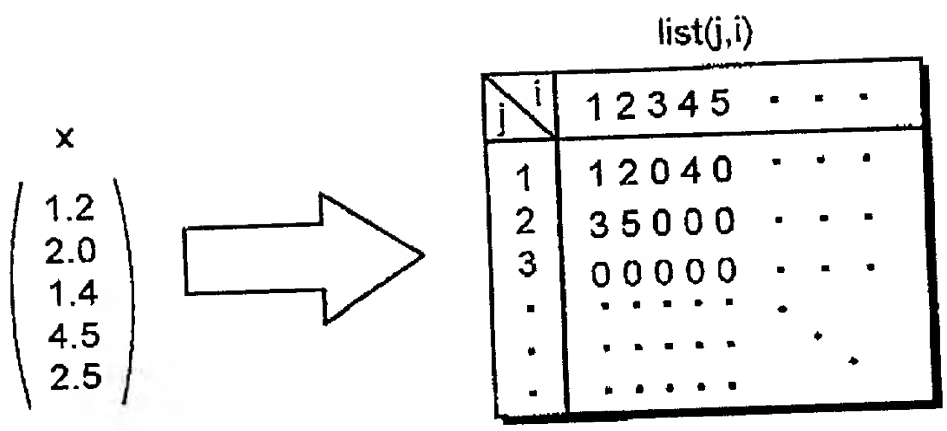

FIG. 13

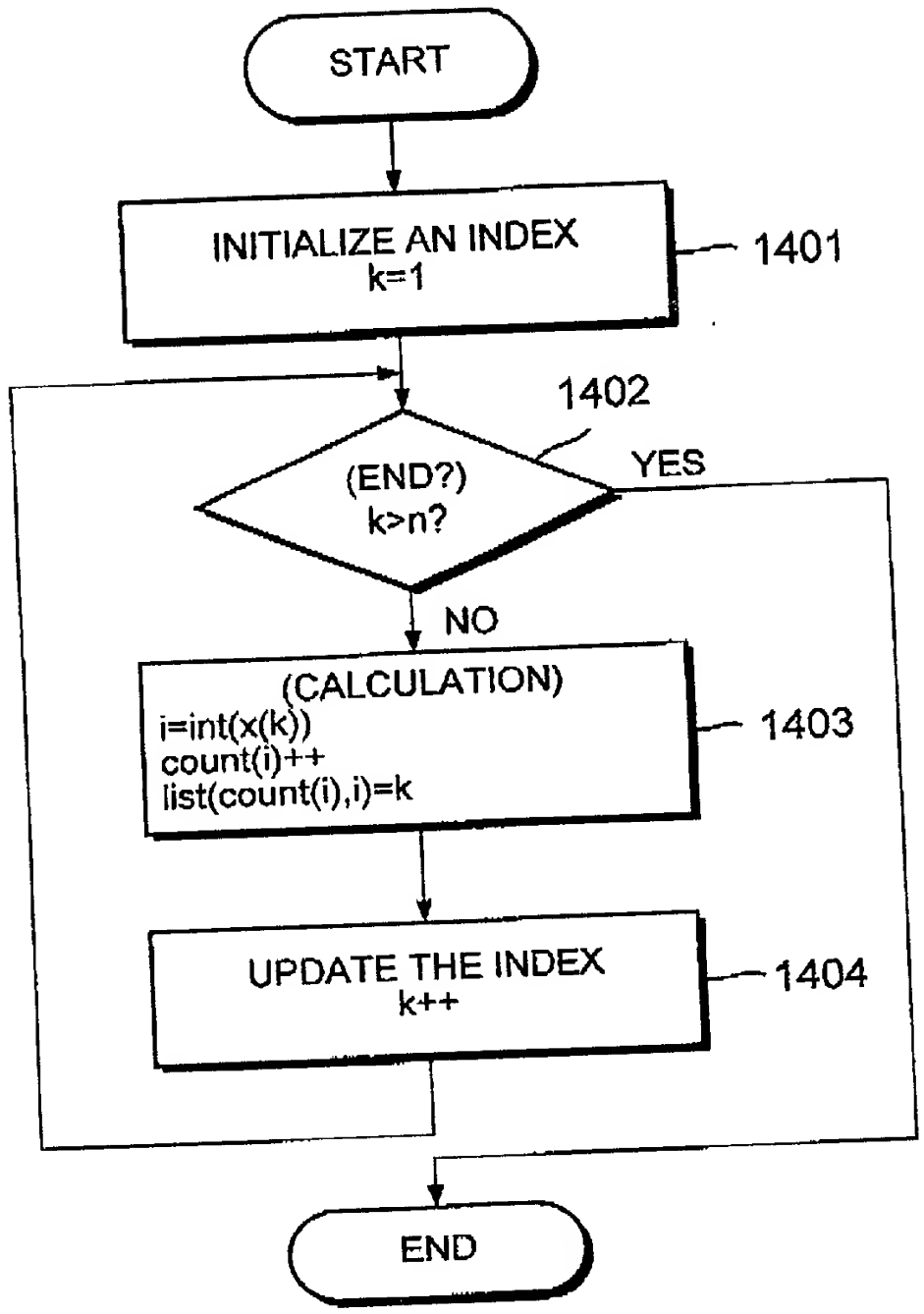

FIG. 14

k	x(k)	int(x(k))	coordi-nates	count(i) 1	2	3	4	5	i/step
1	1.2	1	(1,1)	1	0	0	0	0	1
2	2.0	2	(1,2)	1	1	0	0	0	2
3	1.4	1	(2,1)	2	1	0	0	0	3
4	4.5	4	(1,4)	2	1	0	1	0	4
5	2.5	2	(2,2)	2	2	0	1	0	5
				2	2	0	1	0	

FIG. 15

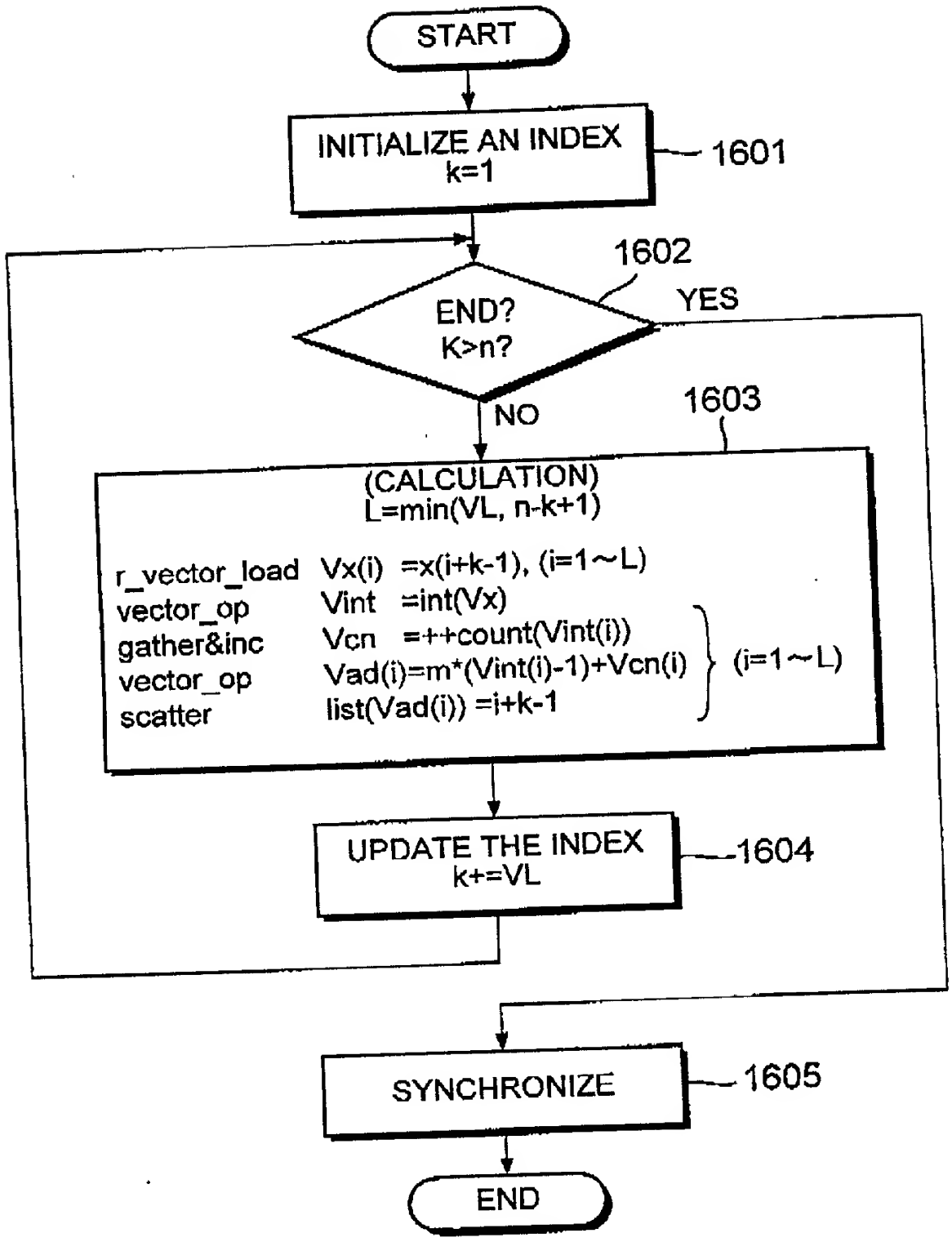
START
INITIALIZE AN INDEX
k=1
1601
1602
END?
K>n?
YES
NO
1603
(CALCULATION)
L=min(VL, n-k+1)
r_vector_load Vx(i) =x(i+k-1), (i=1~L)
vector_op Vint =int(Vx)
gather&inc Vcn =++count(Vint(i))
vector_op Vad(i)=m*(Vint(i)-1)+Vcn(i)
scatter list(Vad(i)) =i+k-1
(i=1~L)
UPDATE THE INDEX
k+=VL
1604
SYNCHRONIZE
1605
END

FIG. 16

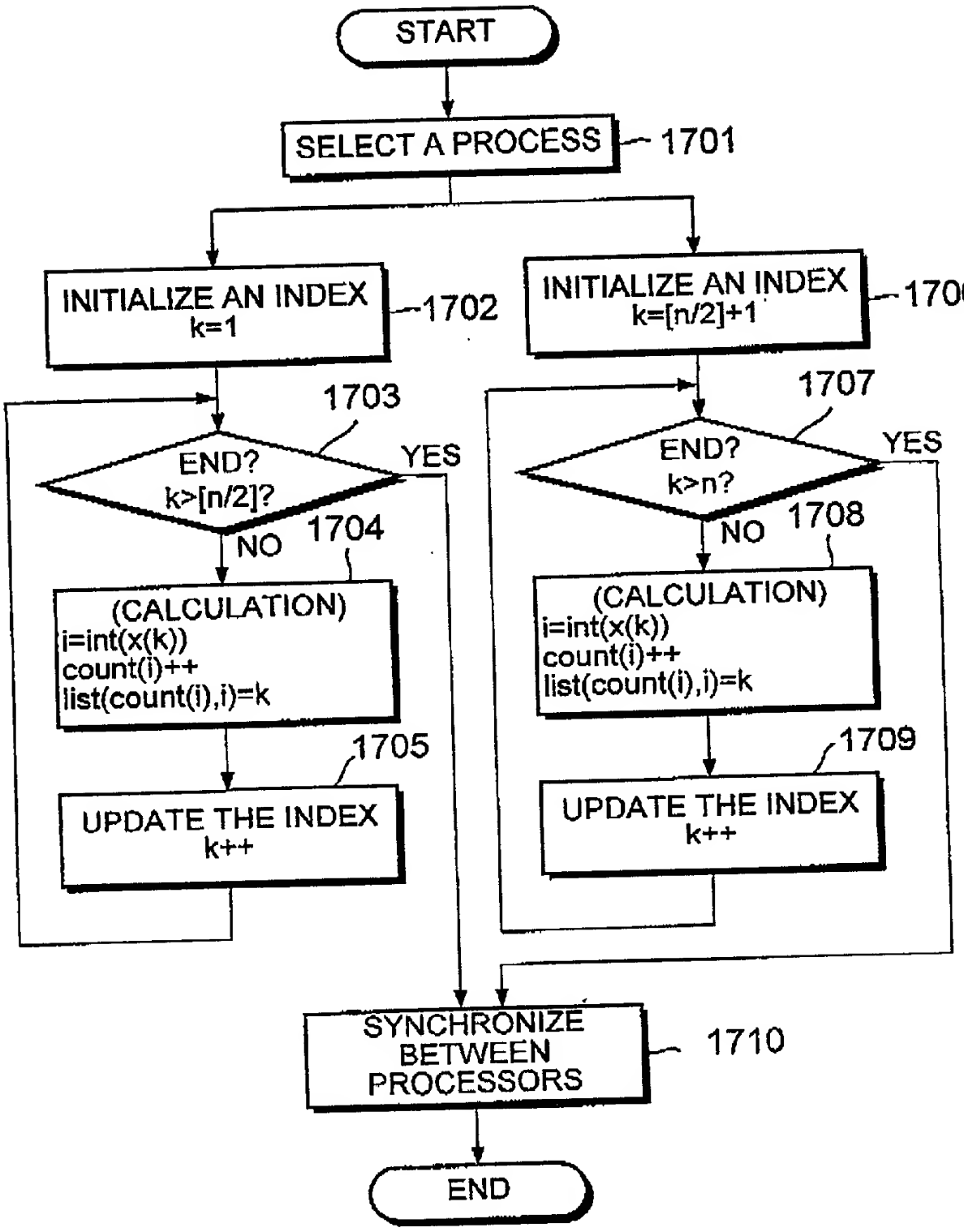
START
SELECT A PROCESS
1701
INITIALIZE AN INDEX
k=1
1702
INITIALIZE AN INDEX
k=[n/2]+1
1706
1703
END?
k>[n/2]?
YES
NO
1704
(CALCULATION)
i=int(x(k))
count(i)++
list(count(i),i)=k
1705
UPDATE THE INDEX
k++
1707
END?
k>n?
YES
NO
1708
(CALCULATION)
i=int(x(k))
count(i)++
list(count(i),i)=k
1709
UPDATE THE INDEX
k++
SYNCHRONIZE BETWEEN PROCESSORS
1710
END

FIG. 17